IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application Serial No. ..... 09/895,954
Filing Date ..... June 29, 2001
Inventorship..... Corbin, L.
Applicant ..... Microsoft Corp.
Group Art Unit ..... 2154
Examiner ..... Nguyen, D.
Attorney's Docket No. ..... 150748.01
Title: Multi-threaded System for Activating a Process Using a Script Engine and Publishing Data Descriptive of the Status of the Process

## APPEAL BRIEF

To: Commissioner for Patents
PO Box 1450
Alexandria, Virginia 22313-1450

From: James R. Banowsky (Tel. 425-705-3539; Fax 425-936-7329)
**Customer No. 22801**

Pursuant to 37 C.F.R. §41.37, Applicant hereby submits an appeal brief for application 09/895,954 filed June 29, 2001, within the requisite time from the date of filing the Notice of Appeal. Accordingly, Applicant appeals to the Board of Patent Appeals and Interferences seeking review of the Examiner's rejections.

## Real Party in Interest

The real party in interest is Microsoft Corporation, the assignee of all right, title and interest in and to the subject invention.

## Related Appeals and Interferences

Appellant is not aware of any other appeals, interferences, or judicial proceedings which will directly affect, be directly affected by, or otherwise have a bearing on the Board's decision to this pending appeal.

## Status of Claims

Claims 1, 3-8, 12-24, 42, 43 and 46-64 stand rejected and are pending in the Application. Claims 1, 3-8, 12-24, 42, 43 and 46-64 are appealed. Some of these claims were previously amended, specifically, claims 1, 3-8, 12-17, 42, 43, 48, 50-53, 57 and 58. Claims 2, 9-11, 25-41, 44 and 45 were previously withdrawn without prejudice. Claims 1, 3-8, 12-24, 42, 43 and 46-64 are set forth in the Appendix of Appealed Claims on page 13.

## Status of Amendments

A Final Office Action was issued on August 5, 2005.

A Response to the Final Office Action requesting reconsideration was filed November 7, 2005. No claims were amended.

An Advisory Action was issued on December 27, 2005, indicating that the request for reconsideration had been considered but did not place the application in condition for allowance.

Appellant filed a Notice of Appeal on February 6, 2006 in response to the Advisory Action and the Final Office Action.

No claims have been amended since the issuance of the Final Office Action.

## Summary of Claimed Subject Matter

A concise explanation of each of the independent claims is included in this Summary section.

**Claim 1** recites a method that enables a system to maintain consistent status information related to an executable process and facilitate the exchange of the information between machines over a distributed network. The information is collected by the system throughout the execution of the process and stored in a retrievable data structure. Any machine having access to the network can communicate with the system and subsequently retrieve the data structure containing the status information.

**Claim 42** describes a system having a node that executes a process management system. The process management system is configured to divide an executing process into threads and to distribute the theads to multiple remote nodes. The process management system collects status information associated with an executing process from each node and stores the information in a data structure, which is accessible by the nodes.

The remote nodes are also included in the system and are configured to execute a script configured to provide the status information collected by the process management system.

**Claim 64** describes an apparatus that receives a request from a client to initiate a process, divides the process into multiple threads and distributes the multiple threads to network nodes for execution. The apparatus also polls each node for status information generated by a script running at each node, receives and stores the status information in a data structure, and enables one or more of the network nodes to access the data structure and, hence, the status information.

## Grounds of Rejection to be Reviewed on Appeal

Claims 1-24 and 42-63 stand rejected under 35 U.S.C. §102(e) as being anticipated by U.S. Patent Publication No. 2002/0054630 of Gitner et al. (hereinafter "Gitner").

It is noted that although claims 1-24 and 42-63 are indicated as rejected, claims 2, 9-11, 25-41, 44 and 45 were withdrawn in a previous response. Therefore, in actuality, claims 1, 3-8, 12-24, 42, 43 and 46-64 can be rejected and, hence, are argued in the present appeal. These claims are properly reflected on the Office Action Summary but are incorrect in the Office Action proper.

## Argument

### A. The rejection under 35 U.S.C. §102(e) over Gitner does not meet minimum requirements for a rejection of anticipation because each element of the rejected claims had not been addressed.

Claims 1, 3-8, 12-24, 42, 43 and 46-64 stand rejected under 35 U.S.C. §102(e) as being anticipated by Gitner.

Applicant respectfully submits that the Office has not established a proper anticipation rejection because each and every element of the claims has not been addresses in the Office Action.

#### The §102 Standard

In making out a §102 rejection, the Federal Circuit has stated that "[a] claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." *Verdegall Bros. v. Union Oil Co. of California*, 814, F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987).

Furthermore, "[t]he identical invention must be shown in as complete detail as is contained in the...claim...." *In re Bond*, 910 F.2d 831, 15 USPQ2d 1566 (Fed. Cir. 1990).

Therefore, if a rejection does not identify an element in a reference that corresponds with an element in the rejection claim, it necessarily follows that the rejection is deficient since it cannot meet the requirements outlined above.

Claim 1 includes steps of (briefly):

(1) receiving a request;

(2) identifying nodes;

(3) polling each identified node;

(4) receiving the status information from each of the nodes;

(5) storing the status information in a data structure; and

(6) enabling [a] client to access the status information.

In the rejection of claim 1, the Office has addressed steps of (briefly):

(1) receiving a request;

(2) identifying nodes in a network;

(3) polling each identified node; and

(4) receiving the status information from each of the nodes.

Without addressing whether Gitner anticipates claim 1, the rejection of claim 1 is facially defective because it does not address all the elements recited in claim 1. Specifically, the "storing" (step 4) and "enabling" (step 5) steps recited in claim 1 are not stated to be anticipated by Gitner.

**Claim 42** also includes a "process management system" that is configured to, among other things, "receive the status information associated with the threads from each remote node and store the status information in a data structure accessible by any node with authorized access to the process management system."

As described above, these elements are not addressed in the Office Action in the rejection of claim 1. The Examiner has added nothing to the

rejection of claim 42, instead simply stating that claim 42 "[does] not teach or define any new limitations above claims 1 and 3."

Therefore, the rejection of claim 42 is also facially invalid.

**Claim 64** recites an apparatus that comprises:

means for receiving a request from a client to initiate a process;

means for dividing the process into multiple threads;

means for distributing the threads to multiple nodes in a network for execution;

means for polling each node for status information generated by a script executing in the node;

means for receiving the status information from each of the nodes;

means for storing the status information in a data structure; and

means for enabling any node with authorization to access the status information.

In the rejection of claim 64, the Examiner stands on his rejection of claim 1 and adds nothing new. But the rejection of claim 1 does not address the "means for storing" or the "means for enabling" steps recited in claim 64.

Therefore, the rejection of claim 64 is facially defective for the same reasons described above in regard to claims 1 and 42.

**B. The rejection under 35 U.S.C. §102(e) over Gitner is improper because Gitner does not anticipate each and every element of the claims.**

In addition to the defects of the rejections explained above, Gitner does not anticipate each and every element of the claims.

**Claim 1** requires the step of "receiving a request from a client for status information related to the process...." The Office Action refers to paragraphs [205] and [675] of Ginter to demonstrate anticipation of this particular element.

Ginter paragraph [205] discloses user requests for clearinghouse information, such as additional credit, electronic currency, etc. Paragraph [675] discloses receiving service requests and routing the service requests to appropriate service providers.

The cited excerpts from Ginter do not disclose or anticipate a client request for *status information* that is *related to an executable process*. For this additional reason, claim 1 is allowable over the cited reference.

Furthermore, claim 1 recites "identifying nodes in a network, each of the nodes executing a distributed thread of the process...." The Office Action points to paragraphs [188], [896], [905-907] and [947-948] as anticipating this element. Paragraph [188] deals with accommodating different control schemes applying to different participants in a network. However, no mention is made of identifying nodes that are each executing a different part of a process.

Paragraph [896] discloses that "kernel/dispatcher 522 may poll each sections/circuits within the SPU 500 and emulate an interrupt for them." But this excerpt does not anticipate identifying nodes in a network that are involved in the execution of a single process.

In the only portion of paragraphs [905] through [907] that seems relevant to claim 1, Ginter states that the kernel/dispatcher 522 may periodically poll a power fail bit in a status word. However, this does not disclose or anticipate identifying nodes as required by claim 1.

Paragraphs [947] and [948] of Ginter states that the Authorization Manager/Service Communications Manager may also support secure server communications between SPE 503 and an external node or device. This does not anticipate identifying network nodes that execute a thread of a process.

Claim 1 also recites "polling each identified node for status information associated with the thread executing by the node, the status information generated by a script associated with the process." Ginter does not disclose or anticipate this element.

Ginter describes a polling mode in an apparatus and performing polling by a kernel/dispatcher. (See Ginter, paragraphs 0896 and 0907 and FIG. 13). However, the polling described in Ginter is performed on components within an apparatus, and not polling multiple nodes in a network for status information related to a multi-threaded process. Thus, the polling described by Ginter is not equivalent to the polling recited in claim 1.

For at least the above-identified reasons, applicant respectfully submits that claim 1 is not anticipated by Ginter.

**Claims 3-8 and 12-24** depend from claim 1 and, therefore, include the same elements and limitations recited therein. Claims 3-8 and 12-24 are allowable over the cited reference at least by virtue of that dependency.

**Claim 42** recites:

> A system comprising:
>
> a process management system executing on a primary node in a network, the process management system configured to collect status information associated with a process, the processing management system also configured to divide the process into multiple threads and distribute the threads to multiple remote nodes in the network, the process management system further configured to receive the status information associated with the threads from each remote node and store the status information in a data structure accessible by any node with authorized access to the process management system; and
>
> the remote nodes in the network, each remote node processing at least one of the threads associated with the process and including a script configured to provide the status information collected by the process management system.

As discussed above, Ginter describes components that communicate with one another to control and distribute content, the use of scripts in the operating system code for metering and transaction management, and polling components within an apparatus. However, nothing in Ginter describes distributing threads of a process to multiple nodes and gathering status information associated with the process from these nodes. Accordingly, Ginter fails to disclose or suggest the process management system, the remote nodes, and their interactions, as recited in claim 42.

Accordingly, claim 42 is not anticipate by Gitner and is, therefore, allowable over Gitner.

**Claims 43 and 46-63** depend from claim 42 and, therefore, include the same elements and limitations recited therein. Claims 43 and 46-63 are allowable over the cited reference at least by virtue of that dependency.

**Claim 64** contains elements similar to those recited in claim 1 and is allowable by the same rationale set forth in the discussion regarding claim 1, above.

## Conclusion

The Office's basis and supporting rationale for the § 102(e) rejections is not supported by the disclosure of the cited reference. Applicant respectfully requests that the rejections be overturned and that the pending claims be allowed to issue.

Respectfully Submitted,

Dated: May 5, 2006 By: _____

James R. Banowsky
Microsoft Corporation
Reg. No. 37,773
(425) 705-3539

**CERTIFICATE OF MAILING OR TRANSMISSION**
**(Under 37 CFR § 1.8(a)) or ELECTRONIC FILING**

I hereby certify that this correspondence is being electronically deposited with the USPTO via EFS-Web on the date shown below:

May 5, 2006
Date

_____
Signature

Noemi Tovar
Printed Name

## Appendix of Appealed Claims

1. (Previously Presented) A method for accessing status information related to a process the method comprising:

receiving a request from a client for status information related to the process;

identifying nodes in a network, each of the nodes executing a distributed thread of the process;

polling each identified node for status information associated with the thread executing by the node, the status information generated by a script associated with the process;

receiving the status information from each of the nodes;

storing the status information in a data structure; and

enabling the client to access the status information.

3. (Previously Presented) The method of claim 1, further comprising:

invoking one or more script engines to execute at least one script code that performs at least one action of the process;

handling multiple script threads during the execution of the process.

4. (Previously Presented) The method of claim 3, wherein the one or more script engines are maintained by a process management system that executes on the nodes.

5. (Previously Presented) The method of claim 4, wherein the one or more nodes include a primary node.

6. (Previously Presented) The method of claim 1, further comprising making the data structure available to any node in the network capable of accessing a process management system in a primary node.

7. (Previously Presented) The method of claim 6, wherein the step of polling is performed by the process management system residing on the primary node over an established connection with the identified nodes.

8. (Previously Presented) The method of claim 7, wherein the identified nodes include the primary node.

12. (Previously Presented) The method of claim 1, wherein the step of storing is performed by a process management system executing on a primary node.

13. (Previously Presented) The method of claim 12, wherein the step of storing further includes:

placing the status information relative to the executable process into a private data structure by the process management system on the primary node, wherein the private data structure is accessible to only script threads that are spawned during the execution of the process.

14. (Previously Presented) The method of claim 12, wherein the step of storing further includes:

placing the status information relative to the executable process into a status value data structure that is accessible to any node capable of accessing the process management system executing on the primary node.

15. (Previously Presented) The system of claim 14, wherein the status value data structure comprises data for providing an indication of an event that occurs during the execution of the process.

16. (Previously Presented) The method of claim 1, further comprising:

establishing a connection between a process management system executing on at least one of the nodes and another process management system residing on a primary node, wherein the connection is established by ~~the~~ a script code in execution by the a script engine associated with the at least one node.

17. (Previously Presented) The method of claim 1, further comprising:

establishing a connection between other client nodes and a process management system residing on a primary node, wherein the connection is established from a user interface executing on the other client nodes; and

accessing the process management system from over the established connection by the user interface executing on the other client nodes.

18. (Original) The method of claim 17, wherein the step of establishing includes accepting a command as input by the user interface to establish a connection with the process management system executing on the primary node.

19. (Original) The method of claim 17, wherein the step of accessing includes accepting a command as input by the user interface to invoke the action of the executable process by the process management system from over the established connection.

20. (Original) The method of claim 17, wherein the step of accessing includes accepting a command as input by the user interface to poll the process management system for status information from over the established connection.

21. (Original) The method of claim 17, wherein the user interface receives messages from the process management system over the established connection.

22. (Original) The method of claim 21, wherein the messages contain information that is descriptive of the primary node.

23. (Original) The method of claim 21, wherein the messages contain information that is descriptive of a particular event that occurs during the execution of the process.

24. (Original) The method of claim 21, wherein the messages contain a data structure that is generated as a result of the execution of the script code by the one or more script engines to indicate the status of the executable process.

42. (Previously Presented) A system comprising:

a process management system executing on a primary node in a network, the process management system configured to collect status information associated with a process, the processing management system also configured to divide the process into multiple threads and distribute the threads to multiple remote nodes in the network, the process management system further configured to receive the status information associated with the threads from each remote node and store the status information in a data structure accessible by any node with authorized access to the process management system; and

the remote nodes in the network, each remote node processing at least one of the threads associated with the process and including a script configured to provide the status information collected by the process management system.

43. (Previously Presented) The system of claim 42, further comprising one or more client node each configured with a user-interface, the one or more user interfaces configured to establish a connection over the network with the process management system executing on the primary node, the one or more user interfaces also configured to request the status information from the process management system and to process the status information when the information is received.

46. (Original) The system of claim 42, wherein the one or more user interfaces accept as input commands to establish a connection with the process management system executing on the primary node.

47. (Original) The method of claim 42, wherein the one or more user interfaces accept as input commands to invoke the action of the executable process by the process management system, and sends requests to invoke the action of the executable process to the process management system from over the established connection.

48. (Previously Presented) The system of claim 42, wherein the one or more user interfaces accept as input commands to poll the process

management system for status information, and sends requests to poll the process management system for status information from over the established connection.

49. (Original) The system of claim 42, wherein the one or more user interfaces receive messages from the process management system over the established connection in response to the polling.

50. (Previously Presented) The system of claim 49, wherein the messages contain information that is descriptive of the primary node.

51. (Previously Presented) The system of claim 49, wherein the messages contain information that is descriptive of a particular event that occurs during the execution of the process.

52. (Previously Presented) The system of claim 49, wherein the messages contain a data structure that is generated as a result of the execution of the script code by the one or more script engines to indicate the status of the executable process.

53. (Previously Presented) The system of claim 42, wherein the process management system accepts connection requests from one or more user interfaces operating on one or more nodes associated with the process management system over an established connection.

54. (Original) The system of claim 53, wherein the one or more nodes include the primary node.

55. (Original) The system of claim 42, wherein the process management system receives requests to invoke the action of the executable process from the one or more nodes connected to the process management system.

56. (Original) The system of claim 42, wherein the process management system continuously polls the one or more nodes connected to the process management system to obtain status information related to the executable process.

57. (Previously Presented) The system of claim 42, wherein the process management system stores the information into a public data structure

that is accessible to the one or more nodes capable of establishing a connection with the process management system.

58. (Previously Presented) The system of claim 42, wherein the process management system stores the status information relative to the process into a private data structure that is accessible to only script threads in operation during process execution.

59. (Original) The system of claim 42, wherein the process management system stores the status information relative to the executable process into a status value data structure that is accessible to the one or more nodes having access to the status information.

60. (Original) The system of claim 59, wherein the status value data structure contains data for providing an indication of a particular event that occurs during the execution of the process.

61. (Original) The system of claim 42, wherein the process management system receives requests for status information relative to the executable

process from the one or more nodes connected to the process management system.

62. (Original) The system of claim 42, wherein the process management system sends the public data structure to the one or more nodes in response to the request.

63. (Original) The system of claim 42, wherein the process management system sends the status value data structure to the one or more nodes in response to the request.

64. (Original) An apparatus comprising:

means for receiving a request from a client to initiate a process;

means for dividing the process into multiple threads;

means for distributing the threads to multiple nodes in a network for execution;

means for polling each node for status information generated by a script executing in the node;

means for receiving the status information from each of the nodes;

means for storing the status information in a data structure; and

means for enabling any node with authorization to access the status information.

## EVIDENCE APPENDIX

## RELATED PROCEEDINGS APPENDIX

None.